Assignment5

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

"'# Import libraries library(caret) library(gbm) library(RANN)

Load data and observe

data("scat") str(scat)

Question 1

Set the Species column as the target/outcome and convert it to numeric.

levels(scat$Species)$scat$Species <- ifelse(scat$Species == "bobcat", 1, $ifelse(scat$Species == "coyote", 2, 3))
target <- scat$Species

Question 2

Remove the Month, Year, Site, Location features.

clean <- subset(scat, select = -c(Month, Year, Site, Location)) str(clean)

Question 3

Check if any values are null. If there are, impute missing values using KNN

sum(is.na(clean)) preprocessed <- preProcess(clean, method = c("knnImpute", "center", "scale")) processed <- predict(preprocessed, clean) sum(is.na(processed))

Question 4

Convert every categorical variable to numerical (if needed)

str(processed) # All the categories are already numerical

Question 5

With a seed of 100, 75% training, 25% testing. Build the following models: random forrest, neural network, naive bayes, and GBM

processed$Species <- -as.factor(target)set.seed(100)index <- -createDataPartition(processed$Species, p = 0.75, list = FALSE) training <- processed[index,] testing <- processed[-index,] str(training)

control <- rfeControl(functions = rfFuncs, method = "repeatedcv", repeats = 3, verbose = FALSE) outcomeName <- 'Species' predictors <- names(training)[!names(training) %in% outcomeName] Species_Pred_Profile <- rfe(training[,predictors], training[,outcomeName], rfeControl = control) Species_Pred_Profile str(predictors) predictors <- c("CN", "d13C", "d15N", "Mass")

RandomForest model

set.seed(100) rfModel <- train(training[,predictors], training[,outcomeName], method = 'rf', importance = T) # Model summarization print(rfModel) # Plot variable of performance rf_vi <- varImp(object = rfModel) plot(rf_vi, main = "RandomForest - Variable of Importance") # Confusion matrix rfPredictions <- predict.train(object = rfModel, testing[,predictors], type = "raw") confusionMatrix(rfPredictions, testing[,outcomeName])

Neural Network model

set.seed(100) nnetModel <- train(training[,predictors], training[,outcomeName], method = 'nnet', importance = T) # Model summarization print(nnetModel) # Plot variable of importance # Variable of importance does not plot for me, but works for Dr. Banda nnet_vi <- varImp(object = nnetModel) plot(nnet_vi, main = "Neural Network - Variable of Importance") # Confusion Matrix nnetPredictions <- predict.train(object = nnetModel, testing[,predictors], type = "raw") confusionMatrix(nnetPredictions, testing[,outcomeName])

Naive Bayes model

set.seed(100) nbayesModel <- train(training[,predictors], training[,outcomeName], method = 'naive_bayes', importance = T) # Model summarization print(nbayesModel) # Plot variable of importance nbayes_vi <- varImp(object = nbayesModel) plot(nbayes_vi, main = "Naive Bayes - Variable of Importance") # Confusion Matrix nbayesPredictions <- predict.train(object = nbayesModel, testing[,predictors], type = "raw") confusionMatrix(nbayesPredictions, testing[,outcomeName])

GBM model

set.seed(100) gbmModel <- train(training[,predictors], training[,outcomeName], method = 'gbm') # Model summarization print(gbmModel) # Plot variable of importance varImp(object = gbmModel) plot(varImp(object = gbmModel), main = "GBM - Variable of Importance") # Confusion Matrix gbmPredictions <- predict.train(object = gbmModel, testing[,predictors], type = "raw") confusionMatrix(gbmPredictions, testing[,outcomeName])

Question 6

For the BEST performing models of each, create and display a dataframe that has the following columns: ExperimentName, accuracy, kappa. Sort the dataframe by accuracy

models_df <- data.frame("ExperimentName" = c("RandomForest", "NeuralNetwork", "NaiveBayes", "GBM"), "Accuracy" = c(max(rfModel$results$Accuracy), max(nnetModel$results$Accuracy), max(nbayesModel$results$Accurac max(gbmModel$results$Accuracy)), "Kappa" = c(max(rfModel$results$Kappa), max(nnetModel$results$Kappa), max(nbayesModel$results$Kappa), max(gbmModel$results$Kappa))) # Sort by accuracy decreasing models_df[order(models_df$Accuracy, decreasing = TRUE),]

Question 7

Tune the GBM model using tune length = 20

gbmModel <- train(training[,predictors], training[,outcomeName], method = 'gbm', tuneLength = 20) # Part A # Print the model summary print(gbmModel) # Part B # Plot the models plot(gbmModel)

Question 8

Using ggplot and gridExtra to plot all variable of importance plots into one single plot

library(ggplot2) library(ggsci) library(gridExtra) library(grid) # Configuring plot for GBM gbm_vi <- varImp(object = gbmModel) gbm_vi <- as.data.frame(gbm_vi$importance)$gbm_vi$Labels <- rownames(gbm_vi) rownames(gbm_vi) <- NULL gbm_vi_plot <- ggplot(gbm_vi, aes(x = reorder(Labels, Overall), y = Overall, color = as.factor(Overall))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = Overall)) + labs(x = NULL, y = "Importance", title = "GBM - Variable of Importance") + coord_flip() + theme_dark() + theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D"), plot.title = element_text(face = "bold", hjust = 0.5)) + scale_color_tron() # Configuring plot for RandomForest rf_vi <- as.data.frame(rf_vi$importance)$colnames(rf_vi) <- c("one", "two", "three")rf_vi$Labels <- rownames(rf_vi) rownames(rf_vi) <- NULL # Plot 1 rf_vi_plot1 <- ggplot(rf_vi, aes(x = reorder(Labels, one), y = one, color = as.factor(one))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = one)) + labs(x = NULL, y = "Importance", title = "1") + coord_flip() + theme_dark() + theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D"))

+ scale_color_tron() # Plot 2 rf_vi_plot2 <- ggplot(rf_vi, aes(x = reorder(Labels, two), y = two, color = as.factor(two))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = two)) + labs(x = NULL, y = “Importance”, title = “2”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() # Plot 3 rf_vi_plot3 <- ggplot(rf_vi, aes(x = reorder(Labels, three), y = three, color = as.factor(three))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = three)) + labs(x = NULL, y = “Importance”, title = “3”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() rf_vi_plots <- grid.arrange(rf_vi_plot1, rf_vi_plot2, rf_vi_plot3, ncol = 2, top = textGrob(“RandomForest - Variable of Importance”, gp = gpar(fontface = “bold”))) # Configuring plot for Neural Network nnet_vi <- as.data.frame(nnet_vi$importance)colnames(nnet_vi) <- c("Overall", "one", "two", "three")nnet_vi$Labels <- rownames(nnet_vi) rownames(nnet_vi) <- NULL # Plot 1 nnet_vi_plot1 <- ggplot(nnet_vi, aes(x = reorder(Labels, Overall), y = Overall, color = as.factor(Overall))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = Overall)) + labs(x = NULL, y = “Importance”, title = “Overall”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() # Plot 2 nnet_vi_plot2 <- ggplot(nnet_vi, aes(x = reorder(Labels, one), y = one, color = as.factor(one))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = one)) + labs(x = NULL, y = “Importance”, title = “1”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() # Plot 3 nnet_vi_plot3 <- ggplot(nnet_vi, aes(x = reorder(Labels, two), y = two, color = as.factor(two))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = two)) + labs(x = NULL, y = “Importance”, title = “2”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() # Plot 4 nnet_vi_plot4 <- ggplot(nnet_vi, aes(x = reorder(Labels, three), y = three, color = as.factor(three))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = three)) + labs(x = NULL, y = “Importance”, title = “3”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() nnet_vi_plots <- grid.arrange(nnet_vi_plot1, nnet_vi_plot2, nnet_vi_plot3, nnet_vi_plot4, ncol = 2, top = textGrob(“Neural Network - Variable of Importance”, gp = gpar(fontface = “bold”)))

Configuring plot for Naive Bayes

nbayes_vi <- as.data.frame(nbayes_vi$importance)colnames(nbayes_vi) <- c("one", "two", "three")nbayes_vi$Labels <- rownames(nbayes_vi) rownames(nbayes_vi) <- NULL # Plot 1 nbayes_vi_plot1 <- ggplot(nbayes_vi, aes(x = reorder(Labels, one), y = one, color = as.factor(one))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = one)) + labs(x = NULL, y = “Importance”, title = “1”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() # Plot 2 nbayes_vi_plot2 <- ggplot(nbayes_vi, aes(x = reorder(Labels, two), y = two, color = as.factor(two))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = two)) + labs(x = NULL, y = “Importance”, title = “2”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() # Plot 3 nbayes_vi_plot3 <- ggplot(nbayes_vi, aes(x = reorder(Labels, three), y = three, color = as.factor(three))) + geom_point() + geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = three)) + labs(x = NULL, y = “Importance”, title = “3”) + coord_flip() + theme_dark() + theme(legend.position = “none”, panel.background = element_rect(fill = “#2D2D2D”)) + scale_color_tron() nbayes_vi_plots <- grid.arrange(nbayes_vi_plot1, nbayes_vi_plot2, nbayes_vi_plot3, ncol = 2, top = textGrob(“Naive Bayes - Variable of Importance”, gp = gpar(fontface = “bold”))) # Plot all variable of importance plots grid.arrange(gbm_vi_plot, rf_vi_plots, nnet_vi_plots, nbayes_vi_plots, ncol = 2, nrow = 2)

Question 9

Which model performs the best? Why do you think this is the case? Can we accurately predict species on this dataset?

Naive Bayes.

The Naive Bayes model predicts the species with 76.94% accuracy and relies on the top 4 predictors more than any other model.

Yes. Although, an accuracy of 90% or more would be ideal.

‘‘‘

Including Plots

You can also embed plots, for example:

```
main = "RandomForest - Variable of Importance")
# Plot variable of importance
# Variable of importance does not plot for me, but works for Dr. Banda
plot(nnet_vi,
main = "Neural Network - Variable of Importance")
plot(nbayes_vi,
main = "Naive Bayes - Variable of Importance")
plot(varImp(object = gbmModel),
     main = "GBM - Variable of Importance")
models_df[order(models_df$Accuracy, decreasing = TRUE),]
gbm_vi <- varImp(object = gbmModel)
gbm_vi <- as.data.frame(gbm_vi$importance)
gbm_vi$Labels <- rownames(gbm_vi)
rownames(gbm_vi) <- NULL
gbm_vi_plot <- ggplot(gbm_vi, aes(x = reorder(Labels, Overall), y = Overall, color = as.factor(Overall))
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = Overall)) +
  labs(x = NULL, y = "Importance", title = "GBM - Variable of Importance") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D"), plot.title = eleme
  scale_color_tron()
# Configuring plot for RandomForest
rf_vi <- as.data.frame(rf_vi$importance)
colnames(rf_vi) <- c("one", "two", "three")
rf_vi$Labels <- rownames(rf_vi)
rownames(rf_vi) <- NULL
# Plot 1
```

```
rf_vi_plot1 <- ggplot(rf_vi, aes(x = reorder(Labels, one), y = one, color = as.factor(one))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = one)) +
  labs(x = NULL, y = "Importance", title = "1") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
# Plot 2
rf_vi_plot2 <- ggplot(rf_vi, aes(x = reorder(Labels, two), y = two, color = as.factor(two))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = two)) +
  labs(x = NULL, y = "Importance", title = "2") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
# Plot 3
rf_vi_plot3 <- ggplot(rf_vi, aes(x = reorder(Labels, three), y = three, color = as.factor(three))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = three)) +
  labs(x = NULL, y = "Importance", title = "3") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
rf_vi_plots <- grid.arrange(rf_vi_plot1, rf_vi_plot2, rf_vi_plot3, ncol = 2, top = textGrob("RandomFores
# Configuring plot for Neural Network
nnet_vi <- as.data.frame(nnet_vi$importance)
colnames(nnet_vi) <- c("Overall", "one", "two", "three")
nnet_vi$Labels <- rownames(nnet_vi)
rownames(nnet_vi) <- NULL
# Plot 1
nnet_vi_plot1 <- ggplot(nnet_vi, aes(x = reorder(Labels, Overall), y = Overall, color = as.factor(Overal
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = Overall)) +
  labs(x = NULL, y = "Importance", title = "Overall") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
# Plot 2
nnet_vi_plot2 <- ggplot(nnet_vi, aes(x = reorder(Labels, one), y = one, color = as.factor(one))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = one)) +
  labs(x = NULL, y = "Importance", title = "1") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
# Plot 3
nnet_vi_plot3 <- ggplot(nnet_vi, aes(x = reorder(Labels, two), y = two, color = as.factor(two))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = two)) +
```

```
  labs(x = NULL, y = "Importance", title = "2") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
# Plot 4
nnet_vi_plot4 <- ggplot(nnet_vi, aes(x = reorder(Labels, three), y = three, color = as.factor(three))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = three)) +
  labs(x = NULL, y = "Importance", title = "3") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
nnet_vi_plots <- grid.arrange(nnet_vi_plot1, nnet_vi_plot2, nnet_vi_plot3, nnet_vi_plot4, ncol = 2, top

# Configuring plot for Naive Bayes
nbayes_vi <- as.data.frame(nbayes_vi$importance)
colnames(nbayes_vi) <- c("one", "two", "three")
nbayes_vi$Labels <- rownames(nbayes_vi)
rownames(nbayes_vi) <- NULL
# Plot 1
nbayes_vi_plot1 <- ggplot(nbayes_vi, aes(x = reorder(Labels, one), y = one, color = as.factor(one))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = one)) +
  labs(x = NULL, y = "Importance", title = "1") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
# Plot 2
nbayes_vi_plot2 <- ggplot(nbayes_vi, aes(x = reorder(Labels, two), y = two, color = as.factor(two))) +
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = two)) +
  labs(x = NULL, y = "Importance", title = "2") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
# Plot 3
nbayes_vi_plot3 <- ggplot(nbayes_vi, aes(x = reorder(Labels, three), y = three, color = as.factor(three
  geom_point() +
  geom_segment(aes(x = Labels, xend = Labels, y = 0, yend = three)) +
  labs(x = NULL, y = "Importance", title = "3") +
  coord_flip() +
  theme_dark() +
  theme(legend.position = "none", panel.background = element_rect(fill = "#2D2D2D")) +
  scale_color_tron()
nbayes_vi_plots <- grid.arrange(nbayes_vi_plot1, nbayes_vi_plot2, nbayes_vi_plot3, ncol = 2, top = text
# Plot all variable of importance plots
grid.arrange(gbm_vi_plot, rf_vi_plots, nnet_vi_plots, nbayes_vi_plots, ncol = 2, nrow = 2)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that

generated the plot.